

REMARKS

Claims 1 through 18 are pending in the present application. Claims 1-18 have been rejected. Claims 1, 4, 9, and 18 have been amended. No new matter has been added.

REJECTIONS

Claims 4, 5, 7, 9-14, 16, and 18 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Chandramohan et al. (U.S. Patent No. 6,711,619), in view of Morton ("Reading CGI Data: URL-encoding and the CGI protocol"). Claims 1-3, 6 and 15 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Chandramohan et al. in view of Morton, and in further view of Krintz et al. ("Reducing the overhead of dynamic compilation"). Claims 8 and 17 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Chandramohan et al., in view of Morton, and in further view of French (U.S. Patent No. 6,654,794). It is respectfully submitted that claims 1-18 are patentable for at least the reasons set forth below.

Independent claims 1, 4, 9, and 18 include features that are neither disclosed nor suggested by the prior art, namely, as represented by claim 4 as amended:

A computer-based method for executing an application on a client computer, the application functioning to process file data stored on a remote server computer, the file data stored on the remote server computer being accessible via web protocols, the method comprising:

(a) **accepting a manually specified execute command for an application entirely stored on the client computer, the execute command including a set of parameters, the set of parameters including an identifier corresponding to the file data, wherein the data file is not executable; and**

(b) **executing a procedure corresponding to the execute command, the procedure manipulating the file data on the client computer by the application,**

wherein step (b) includes downloading the file data from the remote server computer to the client computer using the web protocols without executing a manually specified download command, and the manipulation of the file data on the client computer by the application begins before the file

data has been completely downloaded to the client computer,
and

further wherein downloading the file data comprises
transmitting to the remote server computer an identifier of
executable code and at least one parameter used by the
executable code to derive the file data.

Chandramohan et al. purports to disclose a method for executing portions of applications stored on a network (Chandramohan et al., Abstract). The applications are streamed to clients for execution, allowing the clients to execute the applications without storing them locally (Id.). The applications are speculatively streamed in non-sequential ordered chunks, thereby reducing network latency effects associated with the transmission of code chunks (Id.).

Morton purports to disclose a general overview of the CGI protocol (Morton, page 1). Morton includes a description of common methods for transmitting data to a CGI script, including the GET and POST methods, as well as using the query string and path info (Id.). In addition, Morton describes how a web browser packages CGI data using url-encoding (Id.).

Krintz et al. purports to teach a compilation technique using idle cycles in multiprocessor systems to overlap compilation with application execution (Krintz et al., Summary). Compilation is done in a separate thread from the application, to reduce the possibility of delay in execution (Id.). Profile information is used to prioritize methods for background compilation, such that performance-critical methods are invoked as soon as possible (Id.).

French purports to teach a data processing system that that permits a client system to access a remote resource at a server coupled to the client system by a data network (French, Abstract). An operating system receives file system requests from client devices. The operating system identifies the file system requests as corresponding to a remote resource, and routes the request to a file system driver (Id.). The driver then converts the request to one usable by the remote server, such as HTTP (Id.).

However, neither Chandramohan et al., Morton, Krintz et al., nor French alone or in combination, **teach the manipulation of the file data on the client computer by the**

application before the file data has been completely downloaded to the client computer, where the data file is not executable and the application resides entirely on the client computer as required by the independent claim 4. Applicant has amended independent claim 4 to clarify that the application is stored entirely on the client computer, and that the data file is not executable. In the Examiner's response to Applicants 7/18/2005 arguments, the Examiner states that Chandramohan discloses the partially locally stored application downloading data blocks of the application, and that this teaches an application stored at the client computer downloading a data file, and the application manipulating the data file before it has finished downloading. Applicant respectfully submits that Chandramohan teaches a method where applications are stored remotely. When a user of a client device wishes to execute the application, the application begins streaming to the client device where it may begin to partially execute. In contrast, claim 4 teaches a method where the applications are entirely stored on a client device. However, non-executable data files are stored remotely. When the user wishes to use a particular non executable data files, the files are transferred to the client device, and the application may begin to operate on them before they have completely downloaded. This is completely different then what is disclosed in Chandramohan. In Chandramohan, the applications are stored remotely and streamed to the user when requested. Moreover, these application files are not data files, and are necessarily executable.

Similarly, neither Morton, Krintz et al. nor French, teach **the manipulation of the file data on the client computer by the application before the file data has been completely downloaded to the client computer, where the data file is not executable and the application resides entirely on the client computer**. Accordingly, the combination of Chandramohan et al., Morton, Krintz et al., and French cannot possibly render the independent claim 4 invalid. It is therefore respectfully submitted that the Examiner withdraw the rejections and allow claim 4.

Independent claims 1, 9, and 18 contain similar limitations as independent claim 4, and are therefore allowable for at least the reasons given above for claim 4. It is therefore respectfully requested that the Examiner withdraw the rejections and allow claims 1, 9, and 18.

DOCKET NO.: MSFT-0234
Application No.: 09/767,768
Office Action Dated: October 11, 2005

**PATENT
REPLY FILED UNDER EXPEDITED
PROCEDURE PURSUANT TO
37 CFR § 1.116**

Dependent claims 2, 3, 5-8, and 10-17, are all variously dependent on independent claims 1, 4, and 9, and are therefore allowable for at least the reasons given above for the independent claims. It is therefore respectfully submitted that the Examiner withdraw the rejections and allow claims 2, 3, 5 through 8, and 10 through 17.

In view of the foregoing amendments and remarks, Applicant submits that the above-identified application is in condition for allowance. Early notification to this effect is respectfully requested.

Date: December 2, 2005



Michael W. Tieff
Registration No. 57,845

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439